On the Joint Source-Channel Decoding of Variable-Length Encoded Sources: The BSC Case

K. P. Subbalakshmi and Jacques Vaisey

***Abstract*—This paper proposes an optimal maximum *a posteriori* probability decoder for variable-length encoded sources over binary symmetric channels that uses a novel state-space to deal with the problem of variable-length source codes in the decoder. This sequential, finite-delay, joint source-channel decoder delivers substantial improvements over the conventional decoder and also over a system that uses a standard forward error correcting code operating at the same over all bit rates. This decoder is also robust to inaccuracies in the estimation of channel statistics.**

***Index Terms*—Error resilient communication, joint source-channel decoding, MAP decoding, variable length codes.**

I. Introduction

VARIABLE-LENGTH entropy codes are core components in many state-of-the-art signal compression algorithms; however, these techniques result in extreme sensitivity to channel errors. Unfortunately, the fact that there are many ways to parse the received bit-stream into symbols means that traditional methods for joint-source channel coding (JSCC) cannot be applied directly. In this paper, we propose a joint source-channel decoder (JSCD) for binary symmetric channels (BSCs) that is optimal in the sense that it maximizes the *a posteriori* probability of the received sequence. This decoder achieves improved performance by exploiting the memory and pdf shape of the source distribution.

Preliminary results describing our work first appeared in [1] and [2] and readers are referred to [3] for many more details. Related work by Park and Miller [4] and Demir and Sayood [5] study JSCDs when either the number of transmitted bits or symbols are known *a prior*. Our algorithm requires no such assumptions and hence does not make use of this information, if known. Very recently, a soft input/soft output JSCD using the BCJR trellis was proposed for reversible variable-length codes in [6]. Murad and Fuja propose a *concatenated* JSCD [7] where a Huffman code is followed by a convolutional code.

II. The MAP Decoder

We now consider the MAP decoding of an entropy-coded, discrete-Markov source transmitted over a BSC. This decoder requires both the transition probabilities of the source symbols as well as the channel cross-over probabilities in order to compute the MAP transmitted sequence for a given received sequence. Although we focus below on the case where the source is a first-order process, our algorithm is easily generalized to higher order Markov sources, albeit with increased complexity [3].

Let $\mathbf{C}$ denote the set of all possible B-bit sequences of variable-length codewords. The jth such sequence is then denoted by $\mathbf{c}_j^{n(j)} = \{c_{j,i}\}_{i=1}^{n(j)}$, where $c_{j,i}$ is the codeword corresponding to the ith symbol in the transmitted stream and $n(j)$ is the total number of codewords in this sequence. Similarly, $r_{j,i}$ is the ith symbol of the received bit stream under the same partition as $\mathbf{c}_j^{n(j)}$. The task of the decoder is to take the received bit stream and to search through the members of $\mathbf{C}$ for the most probable transmitted sequence, which we denote by index $\hat{j}$: the probability of receiving $\mathbf{c}_j^{n(j)}$ is $\Lambda(\mathbf{c}_j^{n(j)})$. We note that different partitions of the received bit stream may, in general, lead to different numbers of codewords in the index sequence. More formally, if we define the probability of transmitted symbol s_i to be $\Pr(s_i)$ with $\Pr(s_i \mid s_k)$ representing the probability that s_i is sent immediately after s_k, then

$$\begin{aligned}\Lambda\left(\mathbf{c}_j^{n(j)}\right) &= \Pr(c_{j,1})\epsilon^{H[c_{j,1},r_{j,1}]}(1-\epsilon)^{(l_1 - H[c_{j,1},r_{j,1}])} \\ &\quad \times \prod_{k=2}^{n(j)} \Big[\Pr(c_{j,k} \mid c_{j,(k-1)})\epsilon^{H[c_{j,k},r_{j,k}]} \\ &\quad \times (1-\epsilon)^{(l_k - H[c_{j,k},r_{j,k}])}\Big] \\ \mathbf{c}_{\hat{j}}^{n(\hat{j})} &= \arg\max_{\mathbf{c}_j^{n(j)}} \Lambda\left(\mathbf{c}_j^{n(j)}\right) \qquad (1)\end{aligned}$$

where $H[c_{j,k}, r_{j,k}]$ is the Hamming distance between the kth transmitted codeword of $\mathbf{c}_j^{n(j)}$ and the kth word of the received sequence, and l_k is the length of these words in bits.

A. The State-Space and the Algorithm

To deal with the problem of the decoder not knowing how to parse the bit stream, we propose a novel state-space for the MAP decoder that consists of two classes of states: the *complete* and the *incomplete* states. The decoder is said to be in a complete state if the most recently received bit completed a codeword; otherwise, it is said to be in an incomplete state. In order to make distinctions between the "amount" of incompleteness, we also define the *degree* of an incomplete state to be the number of bits that have already been received in an incomplete codeword.

The number of degrees of incompleteness and the number of transitions between the states are determined by the lengths of the codewords in the codebook. For a codebook

Paper approved by E. Ayanoglu, the Editor Communication Theory and Coding Application of the IEEE Communications Society. Manuscript received July 3, 2000; revised April 26, 2001. This work was supported by the Natural Science and Engineering Research Council of Canada and the Canadian TeleLearning-NCE.

K. P. Subbalakshmi is with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: ksubbala@stevens-tech.edu).

J. Vaisey is with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: vaisey@cs.sfu.ca).

Publisher Item Identifier S 0090-6778(01)10604-5.

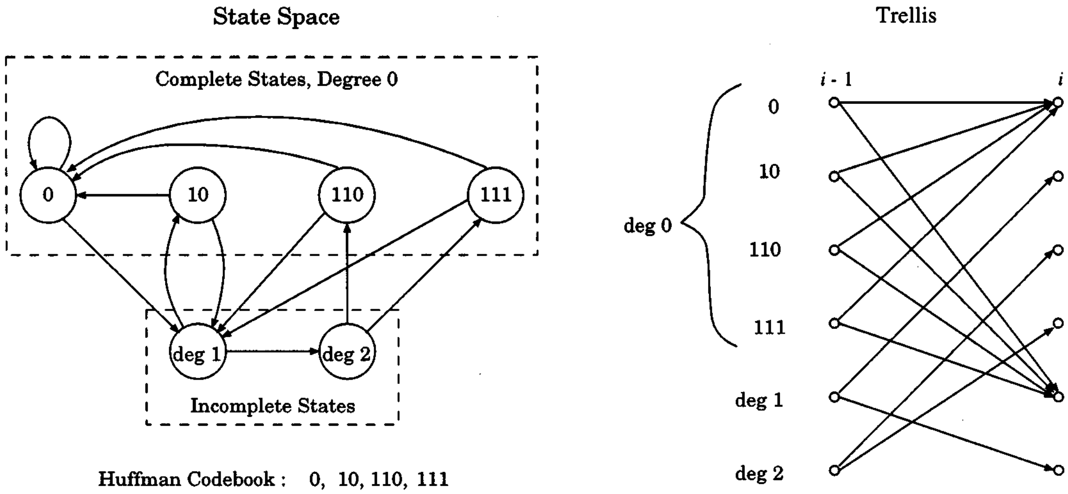

Fig. 1. MAP: example state-space and trellis.

containing N codewords with lengths belonging to the set $\mathcal{L} = \{l_{\min}, \ldots, l_{\max}\}$, the maximum degree of incompleteness is $l_{\max} - 1$ and the total number of states is $N + l_{\max} - 1$. Fig. 1 shows the state-space for an example codebook together with the possible transitions between stages $i - 1$ and i. Each node in the trellis represents a state-stage pair. In the memoryless case, a single degree 0 state can be used to represent all codewords.

The MAP algorithm has two fundamental operations: an examination of the metrics of the paths entering each trellis-node and a search for path merges, with a declaration of decoded symbols when merges are found. For complete states, the path-update step consists of finding the highest metric path to the state and retaining it. For the incomplete states, we must retain the metric values of all the paths back to the last complete state. The complexity of a trellis algorithm is generally considered to be equal to the size of the state-space: $N + l_{\max} - 1$. However, as the incomplete states involve only a copy operation, the complexity is actually somewhat less. Also, when the greatest common divisor of the codeword lengths g is greater than one, further reductions are possible, since we need only consider g samples at a time. Please see [3] and [2] for a formal description of the algorithm.

III. Experimental Results

We define two measures of performance for the algorithms: the percentage of bits that are out of synchronization (PBOS) and the modified signal-to-noise ratio (MSNR). The PBOS captures the synchronization-loss aspect of the performance and is defined as the ratio of the number of bits that are received out of synchronization to the total number of bits in the stream. The MSNR is simply the SNR between the original, unquantized, source and a sequence synthesized from the decoded stream. Specifically, we identify the segments of the recovered sequence with synchronization problems and then force the symbol counts in this segment to be the same as in the corresponding transmitted segment through truncation or zero-padding at the tail end of this decoded segment. These performance measures are appropriate for signals where the synchronization loss is less annoying and, generally, performance measures could be application-dependent. Although we present results for Huffman codes, our decoder is applicable to any entropy-code that can be implemented as a table look-up algorithm.

Experiments were performed on 5000 samples of a zero-mean, first-order Gauss–Markov source using a range of correlation coefficient values, ρ_s; the underlying Gaussian source had unit variance. The test sources were quantized using a 9-level uniform quantizer that provided a "reasonable" SNR under noiseless conditions at an average rate of 3 b/sample for the final Huffman coded stream. The "transmitted" bits were then corrupted with random error patterns to simulate the BSC at different error rates (ϵ_o). For the experiments with conventional forward error correcting codes (FECs), we used the rate $2/3$ convolutional code with constraint length 2, as in [8]. To keep the overall bit rate the same, we code the source at 2 b/sample using a 9-level quantizer and Huffman codes.

Fig. 2 shows the MSNR and PBOS plots with four curves each. These curves correspond to the MAP decoder and Huffman decoder with and without FEC and are labeled accordingly. From the MSNR curves, it can be seen that the MAP decoder for the unprotected Huffman code does better than all others for almost the entire range of ϵ_o, peaking to 8.3 dB over the unprotected Huffman decoder. For the schemes involving FEC, it can be seen that the MAP decoder does better than the Huffman decoder again, as expected, with smaller overall improvement (maximum about 2 dB). The PBOS plot, however, shows a slightly different trend. For $\epsilon_o \in [10^{-3.0}, 10^{-1.2}]$, the FEC protected schemes do better and for $\epsilon_o \in (10^{-1.2}, 10^{-0.7})$, the MAP decoder in the unprotected scheme does better. For very high error rates, the Huffman decoder for the unprotected scheme does better than all others in terms of both MSNR and PBOS. This result is probably due to the fact that the MAP decoders are designed to minimize the probability of sequence errors rather than to optimize the MSNR or PBOS. Finally, we note that the SNR was also calculated for all cases and it was found that the MAP decoder for the unprotected scheme performed better than all others for very low error rates, and

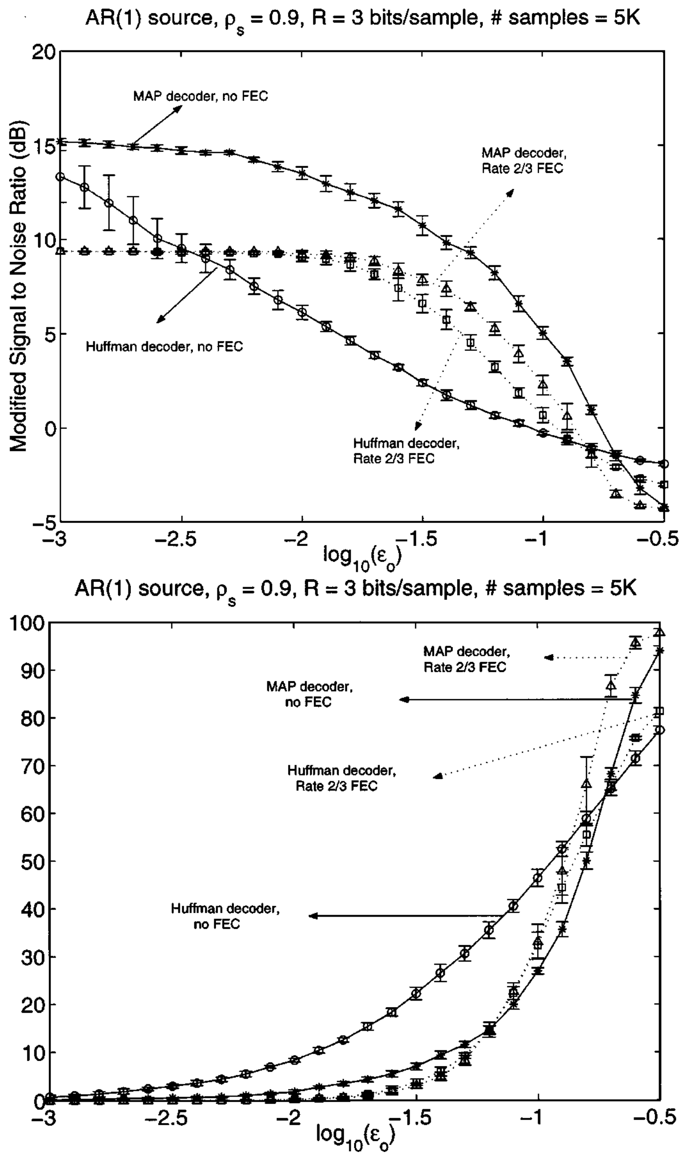

Fig. 2. MSNR and PBOS comparisons for $\rho_s = 0.9$.

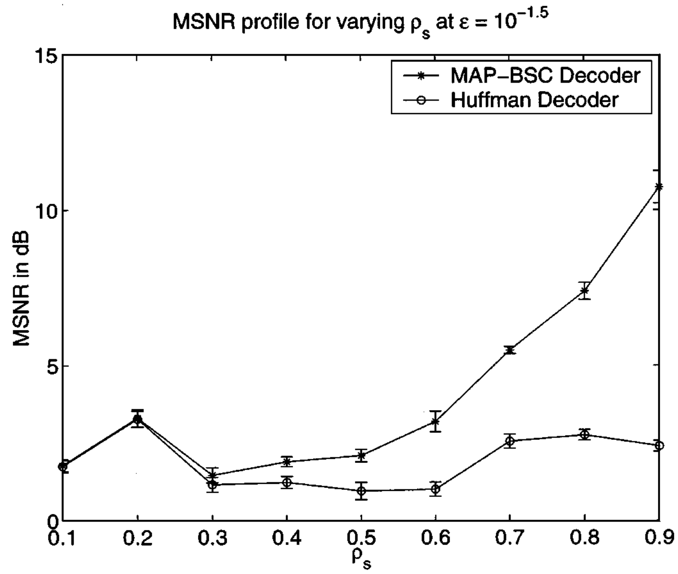

Fig. 3. MSNR comparisons for varying ρ_s at $\epsilon = 10^{-1.5}$.

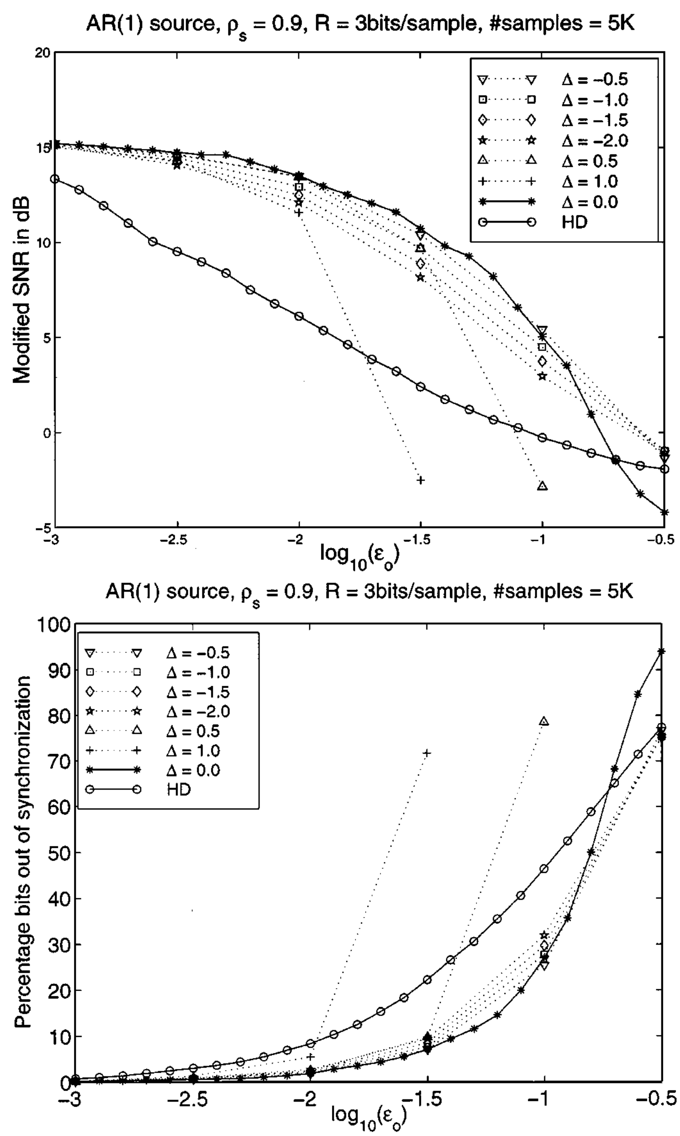

Fig. 4. MAP performance under mismatch, $\rho_s = 0.9$.

at all other error rates the MAP decoder with the FEC scheme did better.

Fig. 3 shows the performance as a function of the source memory ρ_s at a fixed error rate $\epsilon_o = 10^{-1.5}$; as expected, the MAP improvement decreases with ρ_s, approaching zero for $\rho_s < 0.3$. This result is expected, since the residual redundancy available to the decoder drops with ρ_s. The PBOS plot is not shown, since the trend is essentially the same. More experimental results can be found in [3].

In practice, the channel error-rate estimate (ϵ_a) may be poor; however, it turns out that the decoder is quite robust to these mismatches. As a demonstration, we consider the MAP decoder designed for the $\rho_s = 0.9$ case (Fig. 2) and show how the MSNR varies with different mismatch values $(\Delta = \log_{10}(\epsilon_a) - \log_{10}(\epsilon_o))$ in Fig. 4. It is observed that the performance degradation is well below 3 dB when ϵ_a and ϵ_o are within two orders of magnitude.

We also note that, at very high error rates, the MAP operating under a negative Δ mismatch does better than both the $\Delta = 0$

(MAP) and Huffman decoders. It is thus potentially possible to remove the "cross-over" problem observed in Fig. 2 with a judicious choice of Δ! We do not yet have an explanation for this phenomenon; however, we would like to reiterate that the MAP decoder does not directly optimize either of the performance measures considered here, but rather minimizes the probability of error for the entire sequence.

IV. Conclusion

This letter applies a novel state-space structure to design a MAP decoder for variable-length encoded sources transmitted over binary-symmetric channels: knowledge of the number of transmitted symbols is not assumed. The approach was then compared to the standard Huffman decoder using two customized performance measures that attempt to separate the effects of symbol synchronization from that of simple symbol decoding errors. It was found that the MAP decoders performed better for most error rates. The improvement is proportional to the source memory and the decoders are robust to channel error mismatches. The proposed JSCD scheme was also found to perform better than a standard concatenated system comprised of a Huffman code followed by a convolutional code operating at the same overall rate.

References

[1] K. P. Subbalakshmi and J. Vaisey, "Optimal decoding of memoryless sources over binary symmetric channels," in *IEEE Data Compression Conf.*, Snowbird, UT, 1998, p. 573.

[2] ——, "Joint source-channel decoding of entropy coded Markov sources over binary symmetric channels," in *IEEE Int. Conf. Communications*, vol. 1, Vancouver, BC, Canada, June 1999, pp. 446–450.

[3] K. P. Subbalakshmi, "Joint source-channel decoding of variable-length encoded sources with applications to image transmission," Ph.D. dissertation, School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada, July 2000.

[4] M. Park and D. J. Miller, "Joint source-channel decoding of variable-length encoded data by an exact and approximate map sequence estimation," *IEEE Trans. Commun.*, vol. 48, pp. 1–7, Jan. 2000.

[5] N. Demir and K. Sayood, "Joint source/channel coding for variable length codes," in *IEEE Data Compression Conf.*, Snowbird, UT, 1998, pp. 139–148.

[6] R. Bauer and J. Hagenauer, "Iterative source/channel decoding using reversible variable length codes," in *IEEE Data Compression Conf.*, Snowbird, UTh, 2000, pp. 93–102.

[7] A. H. Murad and T. E. Fuja, "Joint source-channel decoding of variable-length encoded sources," presented at the IEEE Information Theory Workshop, June 1998.

[8] J. G. Proakis, *Digital Communications*, 4th ed. New York: McGraw-Hill, 2001.